A dynamic DNA color image encryption method based on SHA-512

entropy

MDPI

Article

A Dynamic DNA Color Image Encryption Method Based on SHA-512

Shihua Zhou [1,2,*], Pinyan He [1] and Nikola Kasabov [2,3]

[1] Key Laboratory of Advanced Design and Intelligent Computing, Ministry of Education, School of Software Engineering, Dalian University, Dalian 116622, China; zhangzhengchin@gmail.com

[2] Knowledge Engineering and Discovery Research Institute, Auckland University of Technology, Auckland 1010, New Zealand; nkasabov@aut.ac.nz

[3] Intelligent Systems Research Center, Ulster University, Londonderry BT52 1SA, UK

* Correspondence: Shihuajo@gmail.com

Received: 2 September 2020; Accepted: 25 September 2020; Published: 28 September 2020

Abstract: This paper presents a dynamic deoxyribonucleic acid (DNA) image encryption based on Secure Hash Algorithm-512 (SHA-512), having the structure of two rounds of permutation–diffusion, by employing two chaotic systems, dynamic DNA coding, DNA sequencing operations, and conditional shifting. We employed the SHA-512 algorithm to generate a 512-bit hash value and later utilized this value with the natural DNA sequence to calculate the initial values for the chaotic systems and the eight intermittent parameters. We implemented a two-dimensional rectangular transform (2D-RT) on the permutation. We used four-wing chaotic systems and Lorentz systems to generate chaotic sequences and recombined three channel matrices and chaotic matrices with intermittent parameters. We calculated hamming distances of DNA matrices, updated the initial values of two chaotic systems, and generated the corresponding chaotic matrices to complete the diffusion operation. After diffusion, we decoded and decomposed the DNA matrices, and then scrambled and merged these matrices into an encrypted image. According to experiments, the encryption method in this paper not only was able to withstand statistical attacks, plaintext attacks, brute-force attacks, and a host of other attacks, but also could reduce the complexity of the algorithm because it adopted DNA sequencing operations that were different from traditional DNA sequencing operations.

Keywords: color image encryption; DNA coding; two rounds of permutation–diffusion; SHA-512

1. Introduction

With the advent of the big data era, numerous digital images, carrying a large amount of information, are generated daily. Accordingly, the security issues of digital images have become increasingly critical. Traditional data encryption algorithms [1], such as RSA, Data Encryption Standard (DES), and Advanced Encryption Standard (AES), however, are not suitable for image encryption because of their large data capacity and the strong correlation between pixel points. Therefore, researchers have begun to look for new solutions for image encryption [1].

As a result of the special characteristics of DNA, excellent parallelism, and large information density, DNA coding [2,3] is popular in image encryption research. Hu et al. [4] proposed DNA-based image cryptography by implementing the DNA cycle operation in the diffusion process, thereby overcoming the limitations of the DNA complementary operation. Chai et al. [5] proposed an image encryption algorithm by employing DNA coding for the diffusion of pixel values. Meanwhile, Liu et al. [6] proposed a remote-sensing image encryption scheme by utilizing a two-dimensional (2D) logistic map to generate DNA masks; this was then used to generate the DNA

matrix. Enayatifar et al. [7] proposed a robust multiple-image encryption with a DNA sequence operation implemented to diffuse the image. Belazi et al. [8] proposed an efficient medical image encryption scheme based on the combination of chaotic systems and DNA computation. Huo et al. [9] proposed a two-round image encryption algorithm utilizing DNA complementary rules. Furthermore, Revathy et al. [10] proposed an authenticated biomedical image transaction based on DNA. Wang et al. [11] used the DNA sequence operation to diffuse the image. Chen et al. [12] proposed a DNA-based image encryption algorithm based on the combination of self-adaptive permutation–diffusion. Liu et al. [13] proposed a color image encryption based on the dynamic DNA and 4-D memresistive hyper chaos. Aashiq et al. [14] presented an image encryption method based on chaotic attractors; on the frequency domain they used the integer wavelet transform to encrypt the image while on the spatial domain they used the DNA sequence. Ballesteros et al. [15] presented a novel method that deviated from traditional schemes, in which variable-length codes based on the Collatz conjecture were used to transform the content of the image into unintelligible audio. Moreover, Ouyang et al. [16] proposed a color image encryption method using the memristive hyperchaotic system and DNA encryption, and Zhu et al. [17] reported an image encryption algorithm based on a matrix of Kronecker products and DNA operations over finite fields. Zhu et al. [18] constructed a five-dimensional continuous hyperchaotic system, and proposed an image encryption scheme based on the hyperchaotic system; this system adopted a dynamic DNA coding mechanism and classical scrambling diffusion encryption structure.

However, some of these DNA-based image cryptography methods pose risks. First, for some DNA-based image encryption schemes, their parameters of the chaotic maps remain unchanged. Second, dynamic DNA coding with different rules is more secure than using only a single rule. Third, a simple confusion or diffusion process is not secure enough. Fourth, an image encryption scheme is not secure enough if the key streams are independent of the plain images.

A secure image encryption scheme should utilize a dynamic permutation and dynamic diffusion process. Moreover, dynamic DNA coding utilizing all rules is more secure than using only a single DNA coding rule. Furthermore, selecting an appropriate chaotic system is also necessary. In addition, the key streams should be dependent of the plain image so that it can resist plaintext attacks. To address these limitations, we proposed a new image encryption algorithm with the structure of two rounds of permutation–diffusion by employing Secure Hash Algorithm-512 (SHA-512), two chaotic systems, dynamic DNA coding, DNA sequencing operations, and conditional shifting.

2. Materials and Methods

2.1. Lorenz System

In 1963, Lorenz tried to explain the unpredictable behavior of the weather by setting up a system of differential equations. In this paper, the image encryption scheme utilizes this system [4]:

$$\begin{cases} c_1^{\cdot} = \alpha(c_2 - c_1) \\ c_2^{\cdot} = \gamma c_1 - c_3 z - c_2 \\ c_3^{\cdot} = c_1 c_2 - \beta c_3 \end{cases} \quad (1)$$

where α, β, and γ are the system parameters. When $\alpha = 10$, $\beta = 8/3$, and $\gamma = 28$, the system is chaotic.

2.2. Four-Wing System

A four-wing system is a four-dimensional hyperchaotic system. The four-wing hyperchaotic system is defined as follows [19]:

$$\begin{cases} x^{\cdot} = ax + byz \\ y^{\cdot} = cy + dz \\ z^{\cdot} = exy + kz + mxw \\ w^{\cdot} = ny \end{cases} \quad (2)$$

where a, b, c, d, e, k, m, and n are the system parameters. When $a = 8$, $b = -1$, $c = -40$, $d = 1$, $e = 2$, $k = -14$, $m = 1$, $n = -2$, its Lyapunov exponents are $LE1 = 1.3938$, $LE2 = 0.5096$, $LE3 = 0$, and $LE4 = -47.8986$. Because there are two positive Lyapunov exponents, the system has hyperchaotic characteristics.

2.3. DNA Coding and Decoding Rule

A DNA sequence consists of four nucleic acid bases, A (adenine), G (guanine), C (cytosine), and T (thymine), which satisfy the Watson–Crick structure [20]. The structure of a DNA sequence is a binary string; on each side of the string, every two nucleic acid bases are complementary, following the rules that A and T are complementary and G and C are complementary. Based on the Watson–Crick structure, only eight combinations can be used for DNA coding [20]. These are listed in Table 1.

TABLE 1. DNA encoding rules.

Rule	1	2	3	4	5	6	7	8
00	A	A	T	T	G	G	C	C
01	C	G	G	C	A	T	A	T
10	G	C	C	G	T	A	T	A
11	T	T	A	A	C	C	G	G

2.4. DNA Complementary Rules

The DNA complementary rule operation is popular for diffusing a DNA matrix. To satisfy the Watson–Crick structure of the DNA sequence, the complementary rules are defined as follows [21]:

$$\begin{cases} x \neq L(x) \neq L(L(x)) \neq L(L(L(x))) \\ x = L(L\left(L(L(x))\right)) \end{cases} \quad (3)$$

In Equation (3), x represents a DNA nucleic acid base. There are six complementary rules [15]: *Rule 1: (AT)(TC)(CG)(GA) Rule 2: (AT)(TG)(GC)(CA) Rule 3: (AC)(CT)(TG)(GA) Rule 4: (AC)(CG)(GT)(TA) Rule 5: (AG)(GT)(TC)(CA) Rule 6: (AG)(GC)(CT)(TA)*

In this paper, the complementary rules are defined as follows:

$$B = DNA\_complementary\_operation(A,times,rules), \quad (4)$$

where A and B are the nucleic acid base before and after the DNA complementary operation, respectively, and $times$ denotes a matrix which indicates how many times the complementary operation is implemented on a nucleic base in the matrix A, and $rules$ denotes a matrix about which rule is chosen for the operation of the DNA complementary operation.

2.5. DNA Cycle Operation

Hu et al. [4] defined another method for DNA matrix diffusion. We use this method in this paper. The DNA cycle function is defined as follows:

$$\text{New nucleic acid base} = L(\text{original nucleic acid base}, h), \quad (5)$$

$$\text{Original nucleic acid base} = L\_1(\text{new nucleic acid base}, h), \quad (6)$$

where L is the function of the DNA cycle operation, and h is how many times the DNA cycle operation is performed on the original nucleic acid base to get the new nucleic acid base.

Figure 1 shows the process of DNA cycle operation and the inverse DNA cycle operation. To explain the Figure 1 in details, for instance, $L(A, 3) = T$, since mod(3, 4) = 3; and $L\_1(A, 7) = G$, since mod(7, 4).

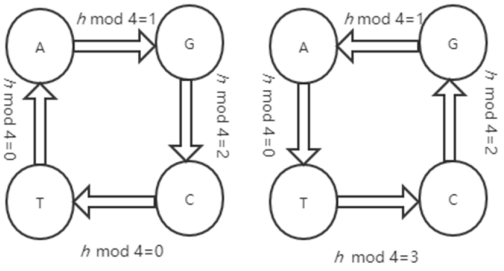


Figure 1. DNA cycle operation.

2.6. Mandelbrot Set

A Mandelbrot set is a plane in which all points belong to a complex plane and whose boundary forms a fractal. The Mandelbrot set is defined as **M**. Set **M** is used for the conditional shifting operation, which is defined later. A typical **M** set is defined as follows [22]:

$$\lim_{n\to\infty} Z_{(n+1)} = Z_n^2 + C \tag{7}$$

where $Z_0 = 0$.

In this paper, a modified Mandelbrot set is defined as follows:

$$W(i,j) = (i \times j) + C, \tag{8}$$

where $W()$ denotes the Mandelbrot set **M**, i = 1,2, …, M and j = 1, 2, …, N, and the size of the image is M × N; C is constant, and C can be any large number. Considering the computational precision on Matlab, in this paper, set C = 10^{14}, which is the most popular choice [22].

2.7. 2D-RT

To solve the limitation of the traditional Arnold maps (i.e., that it cannot permutate the non-square image), this paper used 2D-RT (two-dimensional rectangular transform). The improved 2D-RT can be defined as follows [23]:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \left[\begin{pmatrix} a & b \\ c & d \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} r_m \\ r_n \end{pmatrix}\right] mod \begin{pmatrix} m \\ n \end{pmatrix} \tag{9}$$

and the inverse operation of the improved 2D-RT is expressed as

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1}\begin{pmatrix} x' - r_m \\ y' - r_n \end{pmatrix} mod \begin{pmatrix} m \\ n \end{pmatrix}, \tag{10}$$

where m and n are the sizes of the image. Since 2D-RT was derived from the traditional Arnold map, 2D-RT was an enhanced Tent map and could permutate the non-square image. In this paper, the size of the RGB image P is transformed from $M \times N \times 3$ into $M \times 3N$. 2D-RT is implemented t times to permutate the plain images. In Ref. [23], the system parameters a, b, c, and d satisfy $ad - bc = 1$. In the decryption process, we use the inverse matrix of the original matrix consisting of a, b, c, and d. In the encryption process:

$$PST(x', y') = P(x, y), \tag{11}$$

while in the decryption process:

$$P(x, y) = PST(x', y'). \tag{12}$$

In this paper, P is the plaintext image. In the encryption process, the zero matrix PST with size $M \times 3N$ is defined in previous then the 2D-RT is performed on P for t times to generate the new matrix PST according to Equation (9).

3. Proposed Encryption Scheme

3.1. Initial Values and Intermittent Parameters

In the proposed scheme, SHA-512 is exploited and all the initial values of the chaotic system and the intermittent parameters are generated by the SHA-512 hash function of the plain image.

When the plain image is input, the hash sequence of the plain image with 512 bits is generated: K = [*k1*, *k2*, …, *k64*]. Next, the initial values are generated for the chaotic system.

First, *h1*, *h2*, *h3*, *h4*, *h5*, *h6*, and *h7* are computed as follows:

$$\begin{cases} h1 = \dfrac{k1 + k2 + \cdots + k8}{8 * 256} \\ h2 = 1 + \dfrac{k9 \oplus \ldots \oplus k16}{256} \\ h3 = \dfrac{(k17 \oplus k18 \oplus k19 \oplus k20) + (k21 \oplus k22 \oplus k23 \oplus k24)}{2 * 256} \\ h4 = \dfrac{(k25 \oplus k26 \oplus \ldots \oplus k32)}{256} \\ h5 = \dfrac{(k33 + k34 + k35 + k36) + (k37 \oplus k38 \oplus k39 \oplus k40)}{5 * 256} \\ h6 = \dfrac{k41 + k42 + \cdots + k48}{8 * 256} \\ h7 = \dfrac{(k49 \oplus k50 \oplus \ldots \oplus k56) + (k57 + k58 + \cdots k64)}{9 * 256} \end{cases} \quad (13)$$

Second, one natural DNA sequence is selected, and then it is converted to a decimal number d_s. According to given values of four bases, the corresponding decimals of all bases in the DNA sequence are added. Then, the integer part of the product is removed, and the decimal part is retained. We can get the natural DNA sequence in http://www.ncbi.nlm.nih.gov/ according to geneID, the starting position and the length. For example, we chose a natural DNA sequence with the gene ID of 1054, the starting position of 1022, and the length of 17. The DNA sequence is {TGAAGTTTATACTGTAA}. Then, set A to 0.125112478141254, T to 0.58021545574585, C to 0.98754127451874, and G to 0.96148854586747. The corresponding decimals of all bases in the DNA sequence are added, and the sum is 8.68418997118962. Then, the integer part of 8.68418997118962 is removed, and the decimal part is retained. We can obtain $d_s = 0.68418997118962$. Here, given values, the gene ID, the starting position and the length can all be regarded as part of the key, and they all are set manually.

Next, *h1–h4* defined in Equation (13) and d_s are used to calculate the initial values x_0, y_0, z_0, and ω_0 for the hyperchaotic system, and are is defined as follows:

$$\begin{cases} x_0 = 1 + \dfrac{mod\left((h1 + h2 + d_s) * 10^{14}, 256\right)}{255} \\ y_0 = 1 + \dfrac{mod\left((h2 + h3 + d_s) * 10^{14}, 256\right)}{255} \\ z_0 = 2 + \dfrac{mod\left((h3 + h4 + d_s) * 10^{14}, 256\right)}{255} \\ w_0 = 2 + \dfrac{mod\left((h1 + h2 + h3 + h4 + d_s) * 10^{14}, 256\right)}{255} \end{cases} \quad (14)$$

Meanwhile, *h5–h7* defined in Equation (13) and d_s are used to calculate the initial values c_1, c_2, and c_3 for the Lorenz system:

$$\begin{cases} c_1 = 1 + \dfrac{mod\left((h5 + h6 + d_s) * 10^{14}, 256\right)}{255} \\ c_2 = 1 + \dfrac{mod(h6 + h7 + d_s) * 10^{14}, 256)}{255} \\ c_3 = 2 + \dfrac{mod\left((h5 + h6 + h7 + d_s) * 10^{14}, 256\right)}{255} \end{cases} \quad (15)$$

Finally, the intermittent parameters of $index_1$ to $index_8$ are calculated by the following:

$$\begin{cases} index_1 = mod(k33 + k34 + \cdots + k40,6) + 1 \\ index_2 = mod(k41 + k42 + \cdots + k48,6) + 1 \\ index_3 = mod(k49 + k50 + \cdots + k56,6) + 1 \\ index_4 = mod(k57 + k58 + \cdots + k64,6) + 1 \\ index_5 = mod(k33 \oplus k35 \oplus \ldots \oplus k63,6) + 1 \\ index_6 = mod(k34 \oplus k36 \oplus \ldots \oplus k64,6) + 1 \\ index_7 = mod(k1 \oplus k2 \oplus \ldots \oplus k64,6) + 1 \\ index_8 = mod(k33 \oplus k34 \oplus \ldots \oplus k64,6) + 1 \end{cases} \quad (16)$$

According to Equations (13)–(16), all the initial values of the chaotic systems and the intermittent parameters were determined by the plain image. If there was a one-bit difference between two images, the initial values of the chaotic systems and the intermittent parameters were totally different. Moreover, the chaotic matrices and even the permutated plain images were totally different. Hence, the proposed scheme was sensitive to the plain image.

3.2. Conditional Shifting Operation

In this section, the Mandelbrot set is used for the conditional shifting operation. The conditional shifting operating is defined below Algorithm 1.

Algorithm 1: The Conditional Shifting Operation

Input: Mandelbrot set **M** and the channels R_2, G_2, and B_2.

1: **for** *I = 1:n*
2: find the maximum value of *ith* column elements of **M** and denote it as max_i
3: find the maximum values of the *ith* row elements of R_2, G_2, and B_2 and denote them as max_{ri}, max_{gi} and max_{bi}, respectively, as follows:
4: case 1:
5: **if** $max_i < max_{bi}$, **then**
6: perform left cyclic shift on *ith* elements of R_2 for max_i times
7: else
8: perform right cyclic shift on *ith* elements of R_2 for max_i times
9: **end if**
10: **end**
11: case 2:
12: **if** $max_i < max_{ri}$ **then**
13: perform left cyclic shift on *ith* elements of G_2 for max_i times
14: **else**
15: perform right cyclic shift on *ith* elements of G_2 for max_i times
16: **end if**
17: **end**
18: case 3:
19: **if** $max_i < max_{gi}$ **then**
20: perform left cyclic shift on *ith* elements of B_2 for max_i times
21: **else**
22: perform right cyclic shift on *ith* elements of B_2 for max_i times
23: **end if**
24: **end**
25: **end for**

26: Finally, when the conditional shifting is finished, R_3, G_3, and B_3 are obtained.

3.3. Whole Image Encryption Process

The complete encryption algorithm had a two-round permutation–diffusion structure. In the first round of the permutation–diffusion process, we implemented 2D-RT for permutating the plain image P for t times. Then we decomposed the permutated image P into R_1, G_1, and B_1. The DNA complementary operation was used for the diffusion of the encoded plain image; meanwhile, the DNA cycle operation was implemented for the diffusion of the encoded chaotic matrices. In the second round of the permutation–diffusion process, the conditional shifting was implemented on the decoded images R_2, G_2, and B_2, and we obtained the permutated matrices R_3, G_3, and B_3. Finally, we used the decoded matrices XR, XG, and XB for diffusing matrices R_3 and G_3. The whole encryption process is demonstrated in Figure 2, and the encryption procedures are described in the subsequent subsections.

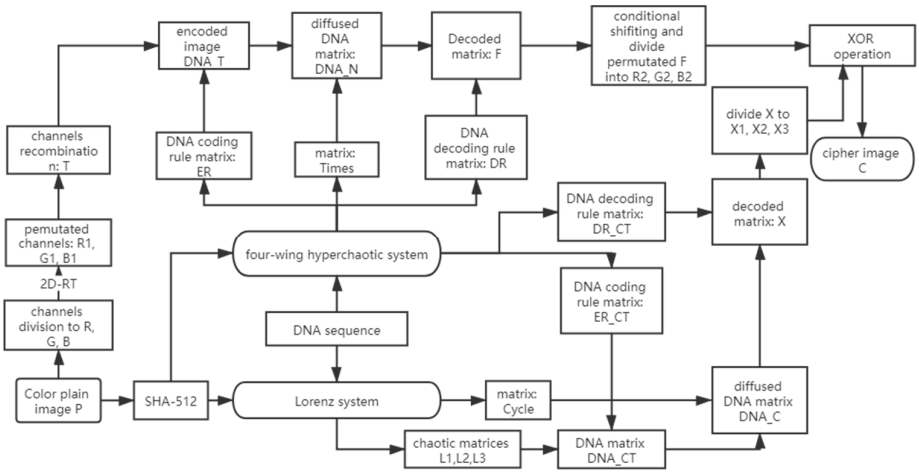


Figure 2. Flowchart of the proposed encryption scheme.

3.3.1. First Round of Permutation

Step 1: Input the RGB plain image $P_{M \times N \times 3}$.

Step 2: Make use of the plain image in the SHA-512 hash function to obtain the initial values for the chaotic systems and the intermittent parameters.

Step 3: Transform the plain image $P_{M \times N \times 3}$ into $P_{M \times 3N}$. Perform 2D-RT on P to permutate the Pt times and obtain the PST.

Step 4: Divide the PST into three channels: R_1, G_1, and B_1.

3.3.2. Process of DNA Encoding

Step 1: Iterate the four-wing chaotic system, with the initial values of x_0, y_0, z_0, and w_0, $4MN + l0$ times. Remove the first $l0$ terms to avoid the transient effect. Four sequences X, Y, Z, and W with the length of $4MN$ are obtained. Next, obtain the sequences X_1, Y_1, and Z_1 by:

$$\begin{cases} X_1 = mod\left(\left(X + Y - fix(X + Y)\right) * 10^{14}, 8\right) + 1 \\ Y_1 = mod\left(\left(Y + Z - fix(Y + Z)\right) * 10^{14}, 8\right) + 1 \\ Z_1 = mod\left(\left(X + Y + Z - fix(X + Y + Z)\right) * 10^{14}, 8\right) + 1 \end{cases} \quad (17)$$

$X_1 = [x_1, x_2, \ldots, x_{4MN}]$, $Y_1 = [y_1, y_2, \ldots, y_{4MN}]$, $Z_1 = [z_1, z_2, \ldots, z_{4MN}]$ are thus obtained.

Step 2: Iterate the Lorenz chaotic system, with the initial values of c_1, c_2, and c_3, $4MN + l0$ times. Remove the first l_0 terms to avoid the transient effect. The three sequences C_1, C_2, and C_3 with the length of $4MN$ are thus obtained. Next, we obtain the sequences L_1, L_2, and L_3:

$$\begin{cases} L_1 = \text{floor}\left(\text{mod}\left(\left(\text{abs}(C_1 + C_2) - \text{floor}(C_1 + C_2)\right) * 10^{14}, 256\right)\right) \\ L_2 = \text{floor}\left(\text{mod}\left(\left(\text{abs}(C_1 + C_3) - \text{floor}(C_1 + C_3)\right) * 10^{14}, 256\right)\right) \\ L_3 = \text{floor}\left(\text{mod}\left(\left(\text{abs}(C_1 + C_2 + C_3) - \text{floor}(C_1 + C_2 + C_3)\right) * 10^{14}, 256\right)\right) \end{cases} \quad (18)$$

Step 3: Convert all the pixels of R_1, G_1, and B_1 into binary numbers, and obtain three $M \times 8N$ matrices *R_bin*, *G_bin*, and *B_bin*. Then, recombine these three matrices into a single matrix T of $3M \times 8N$ by $T = T_i$ (i = 1, 2, …, 6), where $i = index_1$ and

$$T1 = \begin{pmatrix} R\_bin \\ G\_bin \\ B\_bin \end{pmatrix}; T2 = \begin{pmatrix} R\_bin \\ B\_bin \\ G\_bin \end{pmatrix}; T3 = \begin{pmatrix} G\_bin \\ R\_bin \\ B\_bin \end{pmatrix};$$

$$T4 = \begin{pmatrix} G\_bin \\ B\_bin \\ R\_bin \end{pmatrix}; T5 = \begin{pmatrix} B\_bin \\ R\_bin \\ G\_bin \end{pmatrix}; T6 = \begin{pmatrix} B\_bin \\ G\_bin \\ R\_bin \end{pmatrix}.$$

Step 4: Transform the sequence *X1*, *X2* and *X3* into the $M \times 4N$ matrices and transform the sequence *L1*, *L2* and *L3* into the $M \times 4N$ matrices *L_1*, *L_2* and *L_3*.

Step 5: Convert matrices *L_1*, *L_2*, and *L_3* into binary matrices *L1_bin*, *L2_bin*, and *L3_bin* of $M \times 8N$. Then recombine these matrices into a single $3M \times 8N$ binary matrix *CT* by $CT = CTi(i = 1, 2, \ldots, 6)$, where $i = index2$ and

$$CT1 = \begin{pmatrix} L1\_bin \\ L2\_bin \\ L3\_bin \end{pmatrix} CT2 = \begin{pmatrix} L1\_bin \\ L3\_bin \\ L2\_bin \end{pmatrix} CT3 = \begin{pmatrix} L2\_bin \\ L1\_bin \\ L3\_bin \end{pmatrix};$$

$$CT4 = \begin{pmatrix} L2\_bin \\ L3\_bin \\ L1\_bin \end{pmatrix} CT5 = \begin{pmatrix} L3\_bin \\ L1\_bin \\ L2\_bin \end{pmatrix} CT6 = \begin{pmatrix} L3\_bin \\ L2\_bin \\ L1\_bin \end{pmatrix}.$$

Step 6: The parameter $index_3$ is used to construct two DNA encoding rule matrices ER_1 and ER_2 and $ER_1 = ER_{i1}(i1 = 1, 2, \ldots, 6)$, $ER_2 = ER_{i2}(i2 = 1, 2, \ldots, 6)$, $i1 = index_3$, $i2 = mod(index_3, 6) + 1$ and:

$$ER_1 = \begin{pmatrix} X\_1 \\ Y\_1 \\ Z\_1 \end{pmatrix} ER_2 = \begin{pmatrix} X\_1 \\ Z\_1 \\ Y\_1 \end{pmatrix} ER_3 = \begin{pmatrix} Y\_1 \\ X\_1 \\ Z\_1 \end{pmatrix};$$

$$ER_4 = \begin{pmatrix} Y\_1 \\ Z\_1 \\ X\_1 \end{pmatrix} ER_5 = \begin{pmatrix} Z\_1 \\ X\_1 \\ Y\_1 \end{pmatrix} ER_6 = \begin{pmatrix} Z\_1 \\ Y\_1 \\ X\_1 \end{pmatrix}.$$

Step 7: For matrix T, recombine the eight binary planes by combing the first bit plane and the eighth bit plane into the bit plane matrix $T18$, and then do the same to the second bit plane and seventh bit plane, third bit plane and sixth bit plane, and fourth bit plane and fifth bit plane. Through this, we obtained the bit plane matrices $T27$, $T36$, and $T45$. The same operation is performed on the matrix CT, yielding matrices $CT18$, $CT27$, $CT36$, and $CT45$.

Step 8: The encoding rule matrix ER_1 is used to encode matrices $T18$, $T27$, $T36$, and $T45$. $ER_1(1{:}2M,:)$ is used to encode matrix $T18$ and obtain DNA matrix T_DNA18. $ER_1(2M + 1{:}4M,:)$ is used to encode matrix $T27$ to obtain DNA matrix T_DNA27. $ER_1(4M + 1{:}6M,:)$ is utilized to encode matrix $T36$ to obtain DNA matrix T_DNA36. $ER_1(6M + 1{:}8M,:)$ is utilized to encode matrix $T45$ to obtain DNA matrix T_DNA45. Then the four DNA matrices are integrated into a single DNA matrix *DNA_T*: $DNA\_T = [T\_DNA18, T\_DNA27, T\_DNA36, T\_DNA45]$.

The encoding rule matrix ER_2 is used to perform the same operation on matrices $CT18$, $CT27$, $CT36$, and $CT45$; and ER_1 is used for matrices $T18$, $T27$, $T36$, and $T45$. Hence, DNA matrices

*CT_DNA*18, *CT_DNA*27, *CT_DNA*36, *CT_DNA*45, and *CT_DNA* = [*CT_DNA*18, *CT_DNA*27, *CT_DNA*36, *CT_DNA*45] are obtained.

3.4. Diffusion and DNA Decoding

Step 1: Calculate the hamming distance d_1–d_8 by:

$$\begin{cases} d_1 = HD(T\_DNA18, T\_DNA27) \\ d_2 = HD(T\_DNA27, T\_DNA36) \\ d_3 = HD(T\_DNA36, T\_DNA45) \\ d_4 = \dfrac{d_1 + d_2 + d_3}{3} \\ d_5 = HD(CT\_DNA18, CT\_DNA27) \\ d_6 = HD(CT\_DNA27, CT\_DNA36) \\ d_7 = HD(CT\_DNA36, CT\_DNA45) \\ d_8 = \dfrac{d_5 + d_6 + d_7}{3} \end{cases} \tag{19}$$

Update the initial parameters x_0, y_0, z_0, w_0, c_1, c_2, and c_3 by:

$$\begin{gathered} d_1' = \frac{d_1}{3MN}\ d_2' = \frac{d_2}{3MN}\ d_3' = \frac{d_3}{3MN}\ d_4' = \frac{d_4}{3MN}, \\ d_5' = \frac{d_5}{3MN}\ d_6' = \frac{d_6}{3MN}\ d_7' = \frac{d_7}{3MN}\ d_8' = \frac{d_8}{3MN}, \\ x_0' = \frac{x_0 + d_1'}{2}\ y_0' = \frac{y_0 + d_2'}{2} z_0' = \frac{x_0 + d_3'}{2}\, \omega_0' = \frac{z_0 + d_4'}{2}, \\ c_1' = \frac{c_1 + d_1' + d_2'}{2}\ c_2' = \frac{c_2 + d_2' + d_3'}{2} c_3' = \frac{c_3 + d_3' + d_4'}{2}. \end{gathered} \tag{20}$$

Step 2: Utilize the updated initial parameters x_0', y_0', z_0', and w_0' to iterate the four-wing chaotic system $4MN + l1$ times. Remove the first $l1$ times and obtain four sequences X', Y', Z', and W' of $4MN$. Next, use X', Y', and Z' to generate sequences X_2, Y_2, and Z_2:

$$\begin{cases} X_2 = mod(floor\left((X' + Y') - fix(X' + Y')\right) * 10^{14}, 8\right) + 1 \\ Y_2 = mod\left(floor\left((Y' + Z') - fix(Y' + Z')\right) * 10^{14}, 8\right) + 1 \\ Z_2 = mod\left(floor\left((X' + Y' + Z') - fix(X' + Y' + Z')\right) * 10^{14}, 8\right) + 1 \end{cases} \tag{21}$$

Step 3: Convert sequences X_2, Y_2, and Z_2 into matrices *X_2*, *Y_2*, and *Z_2* of $M \times 4N$. Next, use the intermittent parameter $index_4$ and mod ($index_4$, 6) +1 to construct and select the DNA decoding matrix by $DR\_T = DR_{i1}$, $DR\_CT = DR_{i2}$ ($i1$ = 1, 2, …, 6, $i2$ = 1, 2, …, 6), $i1 = index_4$, $i2 = mod$ ($index_4$, 6) + 1 and:

$$DR_1 = \begin{pmatrix} X\_2 \\ Y\_2 \\ Z\_2 \end{pmatrix}\ DR_2 = \begin{pmatrix} X\_2 \\ Z\_2 \\ Y\_2 \end{pmatrix}\ DR_3 = \begin{pmatrix} Y\_2 \\ X\_2 \\ Z\_2 \end{pmatrix}$$

$$DR_4 = \begin{pmatrix} Y\_2 \\ Z\_2 \\ X\_2 \end{pmatrix}\ DR_5 = \begin{pmatrix} Z\_2 \\ X\_2 \\ Y\_2 \end{pmatrix}\ DR_6 = \begin{pmatrix} Z\_2 \\ Y\_2 \\ Z\_2 \end{pmatrix}$$

Step 4: Use the initial parameters c_1', c_2', and c_3' to iterate the Lorenz chaotic system $4MN + l1$ times. Remove the first $l1$ terms to obtain the three sequences C_1', C_2', and C_3' of $4MN$. C_1', C_2', and C_3' are used to obtain the three sequences L_1', L_2', and L_3' by:

$$\begin{cases} L_1' = floor\left(mod\left(\left(abs(C_1' + C_2') - floor(C_1' + C_2')\right) * 10^{14}, 256\right)\right) \\ L_2' = floor\left(mod\left(\left(abs(C_2' + C_3') - floor(C_2' + C_3')\right) * 10^{14}, 256\right)\right) \\ L_3' = floor\left(mod\left(\left(abs(C_1' + C_2' + C_3') - floor(C_1' + C_2' + C_3')\right) * 10^{14}, 256\right)\right) \end{cases} \tag{22}$$

Step 5: Use d_5', d_6', d_7', and d_8' to update the initial parameters x_0', y_0', z_0', and w_0' to obtain x_0'', y_0'', z_0'', and w_0'' by:

$$x_0'' = \frac{x_0'+d_5'}{2}\ y_0'' = \frac{y_0'+d_6'}{2}\ z_0'' = \frac{z_0'+d_7'}{2}\ w_0'' = \frac{w_0'+d_8'}{2}. \tag{23}$$

Then, use the updated initial parameters x_0'', y_0'', z_0'', and w_0'' to iterate the four-wing hyperchaotic system $4MN + l2$ times. Remove the first $l2$ terms and obtain the four sequences X'', Y'', Z'', and W'' of $4MN$. Employ X'', Y'', and Z'' to calculate the new sequences X_3, Y_3, and Z_3:

$$\begin{cases} X_3 = mod(floor\left(\left(X'' + Y''\right) - fix(X'' + Y'')\right) * 10^{14}, 8\right) + 1 \\ Y_3 = mod\left(floor\left((Y'' + Z'') - fix(Y'' + Z'')\right) * 10^{14}, 8\right) + 1 \\ Z_3 = mod\left(floor\left((X'' + Y'' + Z'') - fix(X'' + Y'' + Z'')\right) * 10^{14}, 8\right) + 1 \end{cases} \tag{24}$$

Step 6: Transform sequences W, W', and W'' into $M \times 4N$ matrices W_1, W_2, and W_3, respectively. Then, use the intermittent parameter $index_5$ to construct matrix *Times*, which is used in the DNA complementary operation to determine how many times the operation is performed on a nucleic acid base. $Times = Times_i$ (i = 1, 2, …, 6), $i = index_5$ and:

$$Times_1 = \begin{pmatrix} W\_1 \\ W\_2 \\ W\_3 \end{pmatrix}\ Times_2 = \begin{pmatrix} W\_1 \\ W\_3 \\ W\_2 \end{pmatrix}\ Times_3 = \begin{pmatrix} W\_2 \\ W\_1 \\ W\_3 \end{pmatrix}$$

$$Times_4 = \begin{pmatrix} W\_2 \\ W\_3 \\ W\_1 \end{pmatrix}\ Times_5 = \begin{pmatrix} W\_3 \\ W\_1 \\ W\_2 \end{pmatrix}\ Times_6 = \begin{pmatrix} W\_3 \\ W\_2 \\ W\_1 \end{pmatrix}$$

The final matrix *Times* is calculated by:

$$Times = mod(floor(Times\text{-}fix(Times)) \times 10^{14}, 4) + 1 \tag{25}$$

Step 7: Convert sequences X'', Y'', and Z'' into $M \times 4N$ matrices X_3, Y_3, and Z_3, respectively. Then, use the intermittent parameter $index_6$ to construct and select the complementary rule matrix *CR*, which is used to determine which rule is selected in the DNA complementary operation. $CR = CR_i$ (i = 1, 2, …, 6), $i = index_6$ and:

$$CR_1 = \begin{pmatrix} X\_3 \\ Y\_3 \\ Z\_3 \end{pmatrix}\ CR_2 = \begin{pmatrix} X\_3 \\ Z\_3 \\ Y\_3 \end{pmatrix}\ CR_3 = \begin{pmatrix} Y\_3 \\ X\_3 \\ Z\_3 \end{pmatrix}$$

$$CR_4 = \begin{pmatrix} Y\_3 \\ Z\_3 \\ X\_3 \end{pmatrix}\ CR_5 = \begin{pmatrix} Z\_3 \\ X\_3 \\ Y\_3 \end{pmatrix}\ CR_6 = \begin{pmatrix} Z\_3 \\ Y\_2 \\ X\_1 \end{pmatrix}$$

Step 8: Convert sequences L_1', L_2', and L_3' into matrices L_1', L_2', and L_3', respectively. Then utilize the intermittent parameter $index_7$ for the construction of the matrix *Cycle*, which is used to determine how many times the DNA cycle operation is performed on a nucleic acid base. $Cycle = Cycle_i$ (i = 1, 2, …, 6), $i = index_7$ and:

$$Cycle_1 = \begin{pmatrix} L\_1' \\ L\_2' \\ L\_3' \end{pmatrix}\ Cycle_2 = \begin{pmatrix} L\_1' \\ L\_3' \\ L\_2' \end{pmatrix}\ Cycle_3 = \begin{pmatrix} L\_2' \\ L\_1' \\ L\_3' \end{pmatrix}$$

$$Cycle_4 = \begin{pmatrix} L\_2' \\ L\_3' \\ L\_1' \end{pmatrix}\ Cycle_5 = \begin{pmatrix} L\_3' \\ L\_1' \\ L\_2' \end{pmatrix}\ Cycle_6 = \begin{pmatrix} L\_3' \\ L\_2' \\ L\_1' \end{pmatrix}$$

Step 9: Perform the DNA complementary operation on matrix *T_DNA* to generate matrix *DNA_N:*

$$DNA\_N(i,j) = DNA\_complementary\_operation(T\_DNA(i,j), Times(i,j), CR(i,j)) \quad (26)$$

where $i = 1, 2, \ldots, 3M$ and $j = 1, 2, \ldots, 4N$.

Step 10: Perform the DNA cycle operation on matrix *CT_DNA* to generate matrix *DNA_C*:

$$DNA\_C(i,j) = DNA\_Cylcle\_operation(CT\_DNA(i,j), Cycle(i,j)), \quad (27)$$

where $i = 1, 2, \ldots, 3M$ and $j = 1, 2, \ldots, 4N$.

Step 11: Utilize the DNA decoding matrix *DR_T* to decode DNA matrix *DNA_N*, which is further converted into decimal matrix *F* of $3M \times N$. Meanwhile, utilize DNA decoding matrix *DR_CT* to decode matrix *DNA_C*, which is further converted into decimal matrix X of $3M \times N$.

3.5. Second Round of Permutation and Diffusion

Step 1: Use intermittent parameter $index_8$ to decompose matrix *F* into R_2, G_2, and B_2 of $M \times N$. $F\_1 = F(1{:}M,:)$, $F\_2 = F(M + 1{:}2M,:)$, $F\_3 = F(2M + 1{:}3M,:)$, $i = index_8$, $i = (1, 2, \ldots, 6)$:

$$F_1:\begin{pmatrix} R_2 \\ G_2 \\ B_2 \end{pmatrix} = \begin{pmatrix} F\_1 \\ F\_2 \\ F\_3 \end{pmatrix} F_2:\begin{pmatrix} R_2 \\ G_2 \\ B_2 \end{pmatrix} = \begin{pmatrix} F\_1 \\ F\_3 \\ F\_2 \end{pmatrix} F_3:\begin{pmatrix} R_2 \\ G_2 \\ B_2 \end{pmatrix} = \begin{pmatrix} F\_2 \\ F\_1 \\ F\_3 \end{pmatrix}.$$

$$F_4:\begin{pmatrix} R_2 \\ G_2 \\ B_2 \end{pmatrix} = \begin{pmatrix} F\_2 \\ F\_3 \\ F\_1 \end{pmatrix} F_5:\begin{pmatrix} R_2 \\ G_2 \\ B_2 \end{pmatrix} = \begin{pmatrix} F\_3 \\ F\_1 \\ F\_2 \end{pmatrix} F_6:\begin{pmatrix} R_2 \\ G_2 \\ B_2 \end{pmatrix} = \begin{pmatrix} F\_3 \\ F\_2 \\ F\_1 \end{pmatrix}.$$

Meanwhile, use intermittent parameter *mod* ($index_8$, 6) + 1 to decompose matrix X into XR, XG, and XB. $X\_1 = X(1{:}M,:)$, $X\_2 = (M + 1{:}2M,:)$, $X\_3 = X(2M + 1{:}3M,:)$, $i = mod(index_8,6) + 1$, $i = (1, 2, \ldots, 6)$:

$$\mathrm{X}_1:\begin{pmatrix} X_R \\ X_G \\ X_B \end{pmatrix} = \begin{pmatrix} X\_1 \\ \mathrm{X\_2} \\ \mathrm{X\_3} \end{pmatrix} \mathrm{X}_2:\begin{pmatrix} X_R \\ X_G \\ X_B \end{pmatrix} = \begin{pmatrix} \mathrm{X\_1} \\ \mathrm{X\_3} \\ \mathrm{X\_2} \end{pmatrix} \mathrm{X}_3:\begin{pmatrix} X_R \\ X_G \\ X_B \end{pmatrix} = \begin{pmatrix} \mathrm{X\_2} \\ \mathrm{X\_1} \\ \mathrm{X\_3} \end{pmatrix}$$

$$\mathrm{X}_4:\begin{pmatrix} X_R \\ X_G \\ X_B \end{pmatrix} = \begin{pmatrix} \mathrm{X\_2} \\ \mathrm{X\_3} \\ \mathrm{X\_1} \end{pmatrix} \mathrm{X}_5:\begin{pmatrix} X_R \\ X_G \\ X_B \end{pmatrix} = \begin{pmatrix} \mathrm{X\_3} \\ \mathrm{X\_1} \\ \mathrm{X\_2} \end{pmatrix} \mathrm{X}_6:\begin{pmatrix} X_R \\ X_G \\ X_B \end{pmatrix} = \begin{pmatrix} \mathrm{X\_3} \\ \mathrm{X\_2} \\ \mathrm{X\_1} \end{pmatrix}.$$

Step 2: Calculate the Mandelbrot set **M** by utilizing the introduced method. Use set **M** for the conditional shifting performed on R_2, G_2, and B_2. Finally, R_3, G_3 and B_3 are obtained.

Step 3: Obtain cipher image *C* by:

$$\begin{cases} C(:,:,1) = R_3 \oplus X_R \\ C(:,:,2) = G_3 \oplus X_G \\ C(:,:,3) = B_3 \oplus X_B \end{cases} \quad (28)$$

The proposed cryptosystem was symmetric. We decrypted the encrypted image by applying the encryption in reverse order. Note that we implemented the reverse DNA cycle operation, reverse DNA complementary operation, and reverse 2D-RT instead of the DNA complementary operation, DNA cycle operation, and 2D-RT, respectively. To decrypt the cipher image, the secret keys calculated by the SHA-2 algorithm instead of the hash code calculated by the SHA-2 are transmitted to another user for the decryption of the cipher images.

4. Stimulation Results and Security Analysis

4.1. Stimulation Results

In this section, we conducted stimulation experiments on Windows 7, with 4.00 GB RAM and an i5-4440 CPU. We implemented the scheme in Matlab 2017a (MathWorks, Natick, USA). Images 256×256 in size were used for the encryption and decryption: Lena, Pepper, Baboon, an all-black image, and an all-white image. The three images of objects were in color.

Figure 3a–e are plain images, Figure 3f–j are encrypted images, and Figure 3k–o are decrypted images. As demonstrated in Figure 3, the encrypted images were all noise-like images from which we could not obtain any useful information, but the decrypted images were identical to their plain images, which illustrated that the algorithm was secure and effective.

Additionally, Table 2 shows the system parameters of the 2D-RT, four-wing hyperchaotic system, and Lorenz chaotic system, and the abandoned numbers of the chaotic sequence. We selected one natural DNA sequence (GeneID is 154, and the starting position is 101, and the length is 1213.) to calculate the initial values. Aiming at the natural DNA sequence selected, we set A to 0.125112478141254, T to 0.58021545574585, C to 0.98754127451874, and G to 0.96148854586747. Figure 3a–e are the original images, and Figure 3f–j are the encrypted images corresponding to Figure 3a–e. Figure 3k–o are the decrypted images corresponding to Figure 3f–j.

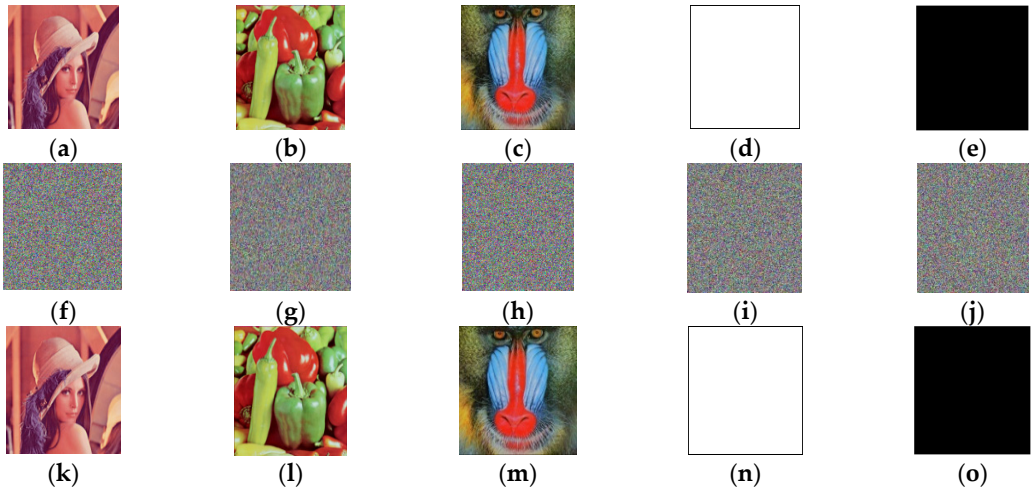


Figure 3. Stimulation results of the proposed scheme.

Table 2. System parameters in the proposed scheme.

Item	Value
System parameters of the four-wing hyperchaotic system	a= 8, b = −1, c = −40, d = 1, e = 2, m = 1, n = −2, n = −14
System parameters of the Lorenz chaotic system	A = 10, β = 8/3, γ = 28
System parameters of the 2D-RT	A = 1, b = 3, c = 5, d = 16, r_m = 4, r_n = 7, t = 5
Abandoned numbers of the sequence	$l0$ = 1000, $l1$ = 1000, $l2$ = 1000

4.2. Key Space Analysis

We used key space analysis to verify the image encryption scheme's ability to resist brute-force attacks. According to [24,25], the key space must be larger than 2^{100} to guarantee the security of the image encryption scheme. In this paper, there are seven initial conditions. If the precision of the computer was 10^{14}, the total key space was $10^{14\times7} = (10^3)^{32.6} \approx (2^{10})^{32.6} = 2^{326}$. The key space of our algorithm is much larger than the theoretical value, so it can resist the exhaustive attack very well. Comparing our key space with others, our key space is also satisfactory. Table 3 shows the comparision of key space. From Table 3, it can be seen that our key space is as good as the key space of others' algorithms, or even better.

Table 3. Comparison of key space.

Algorithm	Ours	Ref. [26]	Ref. [27]	Ref. [28]
Key space	10^{98}	2^{299}	10^{98}	10^{94}

4.3. Key Sensitivity Results

A secure encryption scheme is sensitive to a slight change of the keys. In the proposed encryption scheme, all the keys were generated from the SHA-512 hash function. Therefore, to test the key sensitivity of the proposed encryption scheme, we used the new hash value to change the last bit of the original hash value. The test image was Lena (256 × 256). In this paper, the hash value with the right key was denoted as K (K = 9f63791ec64b3bb5bcf1d6e1272557c9779b37575f33a72e0fbf73a8339bba94d0e3de2ab82ae305ee0a71a122123407227708ff0bc0296768566c2cc59e7d37), with the last bit changed being denoted as K1 (K1 = 9f63791ec64b3bb5bcf1d6e1272557c9779b37575f33a72e0fbf73a8339bba94d0e3de2ab82ae305ee0a71a122123407227708ff0bc0296768566c2cc59e7d38) and the whole new hash value denoted as K2 (K2 = a1100bff91ac78cb8910aafcea1290fc99a3001cbbac73ef31ff23dd1347f90 c60ad23fe26bd4133bad0501a273f0170adfe301261dc3df034ad00ff127526ff). The other encryption keys of the three experiments are the same. GeneID is 154, and the starting position is 22, and the length is 217. Set A to 0.98736273, T to 0.58021545, C to 0.1245737896434, and G to 0.0002356644.

Figure 4a–c show the results obtained upon encrypting Figure 3a with K1, K2, and K, respectively. Figure 5a–c show the results when K was used to decrypt all encrypted images (Figure 4a–c), respectively. Table 4 lists *NPCR* (number of pixels change rate) values between the encrypted images with changed keys and the one with the right key. Table 5 lists *NPCR* values between the decrypted images with changed keys and the original image.

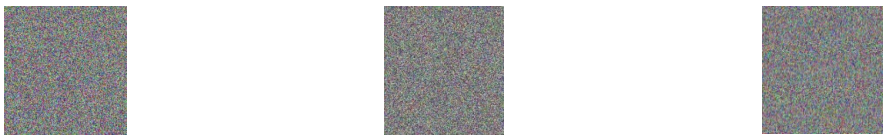
As the figures and tables show, a slight change in the original hash value or a whole new key leads to different encryption and decryption results. In Tables 4 and 5, *NPCR* values with different keys are all close to the expected value of 0.9960, demonstrating that only the complete right hash value generates the right keys that can encrypt and decrypt the images correctly. Therefore, the proposed scheme is sensitive to a slight change in the hash value, which generates totally different keys and leads to totally different encryption and decryption results.

Table 4. NPCR values of the encrypted images.

Image	Changed Key	R	G	B
Figure 4a	K1	0.9963	0.9962	0.9960
Figure 4b	K2	0.9963	0.9957	0.9957
Figure 4c	K	0	0	0

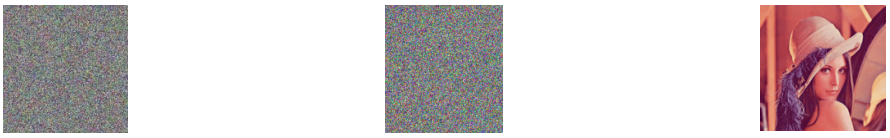
Table 5. NPCR values of the decrypted images.

Image	Changed Key	R	G	B
Figure 5a	K1	0.9965	0.9964	0.9956
Figure 5b	K2	0.9964	0.9962	0.9960
Figure 5c	K	0	0	0



(**a**) (**b**) (**c**)

Figure 4. Key sensitivity results in the encryption process.



(**a**) (**b**) (**c**)

Figure 5. Key sensitivity results in the decryption process.

4.4. Correlation Analysis

Because of the strong correlation among pixels, the traditional encryption scheme could not be directly applied to the images [1]. However, a secure image encryption scheme could eliminate the correlation among pixels. In this section, we used the correlation coefficients to analyze the correlation among pixels between the plain image and encrypted image. Equations (29)–(31) are used to calculate the correlations between pixels in horizontal, vertical and diagonal directions.

Figure 6 shows correlation of two adjacent pixels in the R, G, and B channels for the plain and encrypted image Lena in the horizontal direction. Figure 6a–c show correlations of the original image Lena in the R, G, and B channels respectively, and Figure 6d–f show correlations of the encrypted image Lena in the R, G, and B channels respectively. Figure 6 and Table 6 show that the correlation coefficients of the encrypted images are pretty low, and every pixel distributes evenly. From Table 7, we can see that the proposed scheme is comparable to other schemes in terms of correlation coefficients:

$$r_{x,y} = \frac{E((x-E(x))(y-E(y)))}{\sqrt{D(x)D(y)}}, \tag{29}$$

$$E(x) = \frac{1}{N}\sum_{i=1}^{N} x_i, \tag{30}$$

$$D(x) = \frac{1}{N}\sum_{i=1}^{N}(x_i - E(x))^2. \tag{31}$$

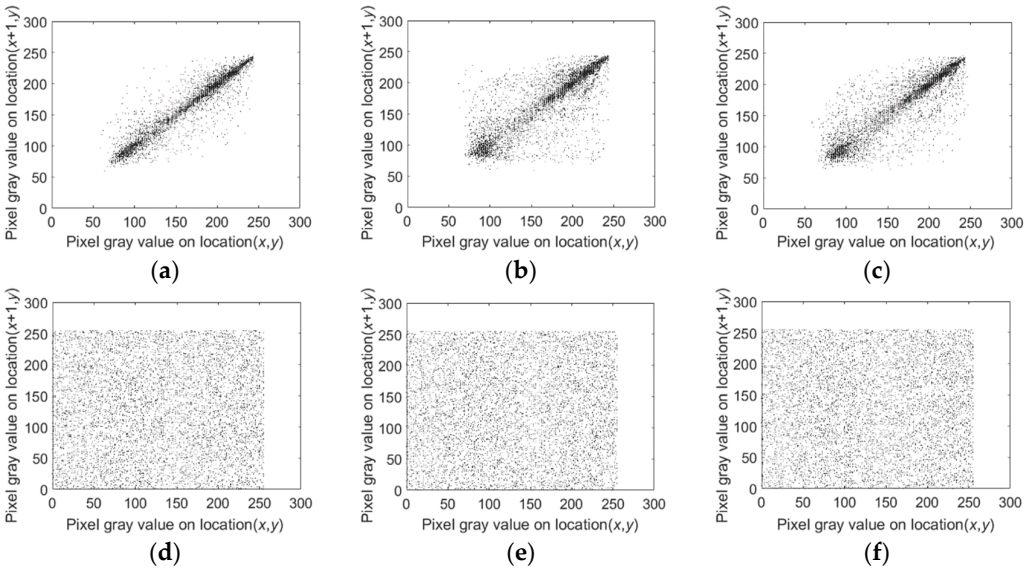


Figure 6. Correlation of two adjacent pixels in the R, G, and B channels for the plain and encrypted image Lena (256 × 256).

Table 6. Correlation coefficients of the plain and encrypted image with the size of 256 × 256.

Image	Direction	Plain Image			Encrypted Image		
		R	G	B	R	G	B
Lena	H	0.968	0.949	0.932	0.014	0.011	0.009
	V	0.943	0.896	0.887	0.011	0.020	0.022
	D	0.918	0.859	0.852	0.035	0.016	0.021
Pepper	H	0.939	0.955	0.925	0.005	−0.015	−0.006
	V	0.931	0.935	0.905	0.029	−0.010	0.011
	D	0.887	0.894	0.842	0.012	−0.014	0.025
Baboon	H	0.917	0.919	0.938	−0.013	0.012	−0.011
	H	0.950	0.895	0.938	−0.014	−0.010	−0.004
	V	0.944	0.876	0.919	−0.022	0.014	−0.007
	D	0.921	0.827	0.889	−0.010	−0.018	0.021
All black	H	#N/A	#N/A	#N/A	−0.011	0.015	0.012
	V	#N/A	#N/A	#N/A	−0.021	−0.016	−0.016
	D	#N/A	#N/A	#N/A	0.016	−0.002	0.005
All white	H	#N/A	#N/A	#N/A	0.001	0.002	0.003
	V	#N/A	#N/A	#N/A	0.005	0.010	0.003
	D	#N/A	#N/A	#N/A	0.003	0.004	0.001

Table 7. Comparison of correlation coefficients across methods.

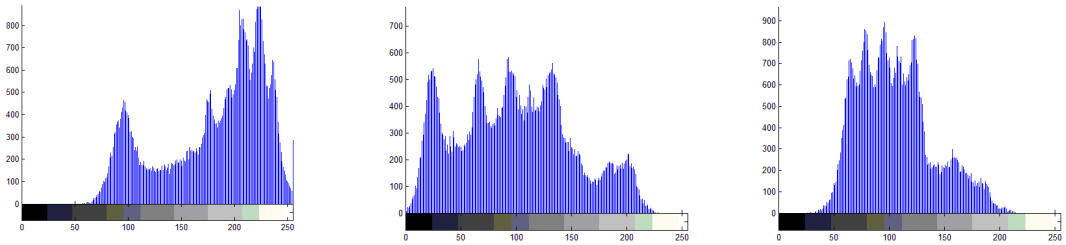
Algorithm	Encrypted Image			
	R	G	B	Average
Ours	0.0011	0.0018	0.0024	0.0018
Ref. [29]	−0.0027	0.0033	−0.0035	0.0031
Ref. [28]	0.0096	0.0109	0.0122	0.0109

4.5. Histogram Analysis

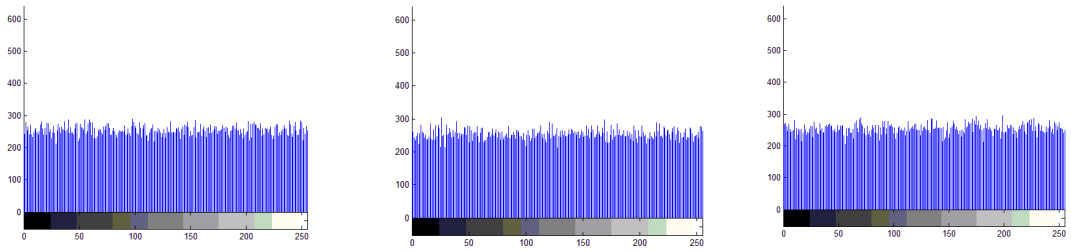
A secure image encryption scheme can resist statistical attacks. The elimination of correlation among pixels was necessary, and pixels of the encrypted image had to be distributed evenly. To verify whether the proposed scheme could distribute the encrypted image evenly, we conducted a histogram analysis.

Figure 7 shows the R, G, and B channels of the plain image Lena and its encrypted image with the size 256 × 256. Table 8 shows the variance of the constructed histograms (calculated by Equation (32)), and Table 9 compares our histograms with those produced by other schemes:

$$var(X) = \frac{1}{n^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{1}{2}(x_i - x_j)^2 \quad (32)$$



(**a**) (**b**) (**c**)



(**d**) (**e**) (**f**)

Figure 7. Histograms of the plain and encrypted image Lena.

From Table 8, we can see that the variances of the original images are very high, while the variances of the encrypted images are greatly reduced. All encrypted variances are reduced by at least 98% compared to the original image variances. Figure 7 shows histograms of the plain and encrypted image Lena. Figure 7a–c are histograms of the plain image Lena in the R, G, and B channels, and Figure 7d–f are histograms of the encrypted image Lena in the R, G, and B channels, respectively, where x-axis denotes the pixel values in the image while y-axis denotes the frequency of the pixels in the image. From Figure 7, the histograms of the original images have obvious peaks, and histograms of the encrypted images are very uniform. Attackers cannot use a statistical attack to obtain any useful information by analyzing the histogram of the encrypted image. Therefore, our method can effectively resist statistical attacks. Table 9 shows comparison of histogram variance across methods about image Lena. It can be seen from Table 9 that our algorithm can obtain encrypted images with lower histogram variance.

Table 8. Histogram data of plain and encrypted images.

Image		Lena	Pepper	Baboon	All Black	All White
Plain image	R	76004.8672	57105.9766	22617.9609	#N/A	#N/A
	G	31563.3516	52138.7656	36848.7813	#N/A	#N/A
	B	95871.8906	103145.2813	35444.8828	#N/A	#N/A
Encrypted image	R	229.5391	259.8532	272.1654	263.6427	238.7628
	G	231.0976	249.9874	276.7468	263.9653	241.7543
	B	247.1986	264.4899	286.8965	255.3785	271.9436

Table 9. Comparison of histogram variance across methods about image Lena.

Algorithm	Variance		
	R	G	B
Ours	229.5391	241.9375	248.1328
Ref. [22]	249.7265	257.4453	256.1875
Ref. [29]	247.7800	279.6200	265.7100

4.6. Information Entropy Analysis

Information entropy is a metric that measures the randomness of an image and the amount of information hidden in an image:

$$H(m) = -\sum_{i=0}^{255} P(x_i) \times log\, P(x_i). \quad (33)$$

Theoretically, a robust encryption scheme has an entropy value of 8. Table 10 shows the information entropy of the plain and encrypted images (size 256 × 256). The entropy is calculated by Equation (33). Our results were very close to 8, and thus were satisfactory. Table 11 compares information entropy across multiple schemes. Our algorithm was superior to other algorithms and was closer to the theoretical value of 8.

Table 10. Information entropy of plain and encrypted images.

Image	Plain Image			Encrypted Image		
	R	G	B	R	G	B
Lena	7.1655	7.5578	6.8571	7.9974	7.9976	7.9975
Pepper	7.3009	7.5570	7.0929	7.9974	7.9973	7.9972
Baboon	7.6987	7.4251	7.5809	7.9970	7.9970	7.9971
All black	0.0000	0.0000	0.0000	7.9971	7.9971	7.9972
All white	0.0000	0.0000	0.0000	7.9974	7.9973	7.9970

Table 11. Comparison of information entropy across methods about image Lena.

Algorithm	Information Entropy		
	R	G	B
Ours	7.9974	7.9976	7.9975
Ref. [29]	7.9973	7.9969	7.9971
Ref. [27]	7.9973	7.9972	7.9969
Ref. [28]	7.9966	7.9972	7.9967

4.7. Differential Attacks and Chosen Plaintext Attack

Differential attacks crack the symmetric encryption scheme by analyzing the information distribution of the encrypted image. A secure symmetric encryption scheme is capable of resisting such attacks.

The *NPCR* and *UACI* (unified average change intensity) values of the R, G, and B channels are calculated as follows:

$$NPCR = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} D(i,j)}{M \times N} \times 100\%, \tag{34}$$

$$UACI = \frac{1}{255 \times M \times N}\left[\sum_{i=1}^{M}\sum_{j=1}^{N} C(i,j) - C'(i,j)\right] \times 100\%, \tag{35}$$

$$D(i,j) = \begin{cases} 0, if\ C(i,j) = C'(i,j) \\ 1, if\ C(i,j) \neq C'(i,j) \end{cases}. \tag{36}$$

Table 12 lists the *NPCR* and *UACI* values of encrypted images with a size of 256 × 256. Table 13 compares these values with those obtained through other schemes. As the tables illustrate, the *NPCR* and *UACI* of the R, G, and B channels were very close to the ideal values of 0.996 and 0.3346, respectively. Furthermore, the values of the proposed scheme were as good as the values obtained by the other methods. A secure and efficient encryption method is sensitive to a slight change in the plain image, and hence the encryption scheme is capable of resisting plaintext attacks. Usually, hackers employ all-black and all-white images to perform the chosen plaintext attack. As seen in Table 12, the NPCR and UACI of all-black and all-white images were close to the ideal values, thereby illustrating that the proposed scheme was sensitive to the plaintext and therefore could resist these attacks.

Table 12. *NPCR* and *UACI* values of different encrypted images.

Image	*NPCR*			*UACI*		
	R	G	B	R	G	B
Lena	0.9959	0.9960	0.9961	0.3354	0.3344	0.3345
Pepper	0.9962	0.9960	0.9959	0.3341	0.3339	0.3336
Baboon	0.9960	0.9961	0.9959	0.3345	0.3340	0.3334
All black	0.9961	0.9961	0.9958	0.3344	0.3345	0.3341
All white	0.9963	0.9959	0.9962	0.3344	0.3334	0.3351

Table 13. Comparison of *NPCR* and *UACI* values across methods about image Lena.

Image	*NPCR*			*UACI*		
	R	**G**	**B**	**R**	**G**	**B**
Ours	0.9959	0.9960	0.9961	0.3354	0.3344	0.3345
Ref. [29]	0.9960	0.9961	0.9961	0.3356	0.3345	0.3349
Ref. [28]	0.9961	0.9961	0.9961	0.3343	0.3343	0.3342
Ref. [30]	0.9963	0.9960	0.9960	0.3360	0.3330	0.3340

4.8. Noise and Occlusion Attack Analysis

During image transmission, noise and occlusion attacks are inevitable, but a robust encryption scheme can resist them. To verify whether the proposed scheme was capable of resisting noise and occlusion attacks, we used the encrypted image of Lena (256 × 256) as the test image. Salt-and-pepper noise (SPN) and Gaussian noise (GN) of varying intensities were added to the test image. In occlusion attack analysis, we added an occlusion effect to the test images; the occluding object occupied different proportions of the image and occurred at different positions.

In addition, we employed the peak-signal-to-noise ratio (*PSNR*) to calculate the difference between the original image and the decrypted images. The *PSNR* is calculated as follows:

$$PSNR = 10 \times \log_{10}\left(\frac{255 \times 255}{MSE}\right) (dB), \tag{37}$$

$$MSE = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}[P_1(i,j) - P_2(i,j)]^2, \tag{38}$$

where M and N are the width and height of an image, respectively, and $P1$ and $P2$ are the original plain image and the image decrypted from the contaminated cipher image, respectively.

Figure 8 shows stimulation results of occlusion attacks with Lena. Figure 8a,c,e and g respectively represent the images obtained after Figure 3f suffered the different occlusion attack. Figure 8b,d,f and h respectively represent the decrypted images of Figure 8a,c,e and g Evidently, according to Figure 8 and Table 14, all PSNR values were larger than 27, and the decrypted images were all recognizable despite the various sorts of contamination in the encrypted images. Therefore, the proposed method was capable of resisting noise attacks and occlusion attacks.

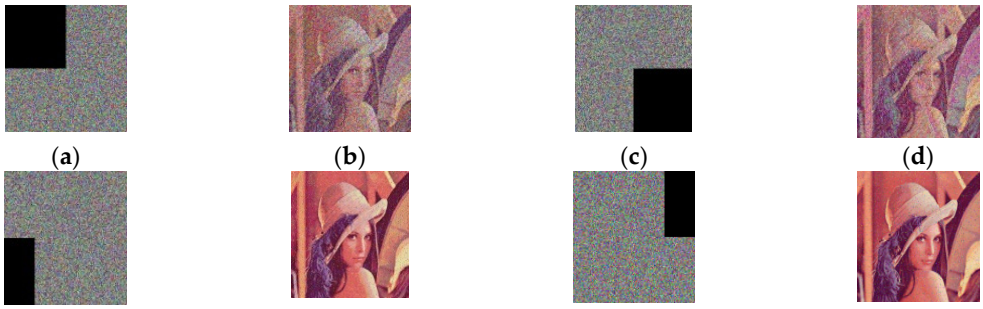


Figure 8. Stimulation results of occlusion attacks with Lena (256 × 256).

Table 14. *PSNR* results with Lena (256 × 256).

Item	R	G	B
GN with intensity = 0.02	28.2541	28.5421	28.3041
GN with intensity = 0.2	27.5014	27.3657	27.4251
SPN with intensity = 0.0002	57.4214	56.3527	56.8765
SPN with intensity = 0.0005	66.5047	67.4581	66.5041
SPN with intensity = 0.001	59.1021	61.1042	61.5384
1/8 data loss at the lower-left corner	30.6874	34.5478	35.6522
1/8 data loss at the upper-right corner	33.0001	33.0487	32.6894
1/4 data loss at the lower-right corner	31.5478	31.2587	31.3586
1/4 data loss at the upper-left corner	29.9564	32.7532	32.2287

4.9. Resistance to Some Typical Attacks

A secure cryptosystem should be capable of resisting cipher-text only attack, chosen-ciphertext attack, known-plaintext attack and chosen-plaintext attack. Among them, the chosen-plaintext attack is the most powerful. And if a cryptosystem is capable of resisting the chosen-plaintext attack, this cryptosystem is capable of resisting three other types of attack and we can declare that this cryptosystem is secure enough. In this paper, the encryption algorithm consists of two rounds of permutation-diffusion. In which, DNA encoding, DNA diffusion operation, DNA decoding, chaos and other techniques are used. And in our algorithm, the SHA-512 algorithm and the natural DNA sequence are used to generate the initial values of two chaotic systems. And the different image leads to the different initial values for the chaotic systems. Evidently, our algorithm is dependent on the plain image directly. In addition, if the hackers use the specific images such as all white and all black images to perform chosen-plaintext attack on our algorithm, the stimulation results of the all-white image and all-black image show that these two images all noise-like ones. Therefore, we can conclude that the proposed algorithm is capable of resisting the above mentioned typical attacks.

4.10. Contrast Investigation

Contrast investigation [31,32] is usually to calculate the local intensity variance in the image. Contrast is luminescence or color difference, through which objects in the image can be distinguished, and because the observer can recognize different objects. A higher contrast value indicates that the image has significantly different gray levels, while a constant gray level is represented by a lower value. Its mathematical description is:

$$C = \sum_{i,j} |i - j|^2 \times p(i,j) \tag{39}$$

where $p(i,j)$ indicates the number of gray-level co-occurrence matices (GLCM).

Table 15 shows contrast values of plain images and encrypted images in R, G, B channels. From Table 15, contrast values of encrypted images are higher than ones of plain images. According to Ref. [31,32], it can prove that our method is satisfactory in terms of comparative investigation.

Table 15. Contrast values of plain images and encrypted images in R, G, B channels.

Image	Plain Image			Encrypted Image		
	R	G	B	R	G	B
Lena	0.3672	0.3947	0.3405	10.5208	10.4763	10.5223
Pepper	0.1743	0.2341	0.1668	10.4999	10.4879	10.5112
Baboon	0.2248	0.2204	0.2430	10.5261	10.5012	10.4987

4.11. Energy

Energy calculations [31,32] result in the addition of square elements in GLCM. When the entries of GLCM are almost equal, the value of energy is lower, and when the amplitude of some entries is higher, the value of energy is higher. For encrypted images, the energy must be low:

$$E = \sum_{i,j} p(i,j)^2 \tag{40}$$

where $p(i,j)$ indicates GLCM.

Table 16 shows energy values of plain images and encrypted images in R, G, B channels. According to Ref. [31,32], Table 16 can illustrate that encrypted images have the lower energy, and our method is satisfactory in terms of energy.

Table 16. Energy values of plain images and encrypted images in R, G, B channels.

Image	Plain Image			Encrypted Image		
	R	**G**	**B**	**R**	**G**	**B**
Lena	0.1391	0.0989	0.1756	0.0156	0.0156	0.0156
Pepper	0.1499	0.1183	0.1849	0.0156	0.0156	0.0156
Baboon	0.1047	0.1285	0.1233	0.0156	0.0156	0.0156

5. Conclusions

In this paper, we proposed an image encryption method with two rounds of permutation and diffusion. First, we employed the SHA-512 algorithm and the natural DNA sequence to generate the initial values for the four-wing hyperchaotic system and the Lorenz chaotic system, and the intermittent parameters. Since the hash value was determined by the plain image, a slight change in the plain image led to a totally different hash value so that the initial values for the chaotic system and the intermittent parameters were totally different, thereby leading to a totally different encrypted image in the end. Therefore, the proposed method was a one-time key pad scheme and was capable of resisting plaintext attacks. Second, we performed 2D-RT on the plain image t times. This was the first round of permutation. Since 2D-RT was derived from the traditional Arnold map, 2D-RT was an enhanced Tent map and could permutate the non-square image. Third, we employed the initial values to generate the chaotic sequences and the chaotic matrices for the construction of the DNA encoding rule matrices. All the DNA encoding rules depended on the plain image. Fourth, we used the intermittent parameters to construct the DNA matrices. Furthermore, the DNA matrices were used to calculate the hamming distances to update the initial values and iterate the chaotic systems for the second time, which eliminated the risk of using the secret keys several times. Fifth, the new chaotic matrices were generated, and the intermittent parameters were used to construct the DNA decoding rule matrices, making all the DNA decoding rules determined by the plain image. All of the rules were used in the first round of diffusion. In contrast to the traditional diffusion operations implemented on DNA matrices, in the proposed scheme, two different DNA diffusion operations were implemented on the encoded plain images and the encoded chaotic matrices: the dynamic DNA complementary rule operation and the DNA cycle operation. Finally, the eighth intermittent parameters were used to decompose the encoded images and encoded chaotic matrices, and in the second round of permutation–diffusion, we performed conditional shifting on the decomposed images and implemented the XOR calculation with the decomposed chaotic matrices to get the final encrypted image. Stimulation results and security analysis illustrated that the proposed scheme was secure and capable of resisting various sorts of attacks, and produced satisfactory stimulation results on image encryption and image decryption.

Author Contributions: Conceptualization, S.Z. and P.H.; methodology, S.Z.; software, P.H.; validation, S.Z., P.H. and N.K.; formal analysis, S.Z.; investigation, P.H.; resources, S.Z.; data curation, S.Z. and P.H.; writing—original draft preparation, S.Z. and P.H.; writing—review and editing, S.Z. and N.K.; visualization,

P.H.; supervision, S.Z. and N.K.; project administration, S.Z. and N.K.; funding acquisition, S.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Key R & D Program of China (No. 2018YFC0910500), the National Natural Science Foundation of China (Nos.61425002, 61751203, 61772100, 61972266, 61802040, 61672121, 61572093), Program for Changjiang Scholars and Innovative Research Team in University (No. IRT_15R07), the Program for Liaoning Innovative Research Team in University (No. LT2017012), the Natural Science Foundation of Liaoning Province (Nos. 20180551241, 2019-ZD-0567), the High-level Talent Innovation Support Program of Dalian City (Nos. 2017RQ060, 2018RQ75), the Dalian Outstanding Young Science and Technology Talent Support Program (No. 2017RJ08), and the Scientific Research Fund of Liaoning Provincial Education Department (No. JYT19051).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Huang, H.; He, X.; Xiang, Y.; Wen, W.; Zhang, Y. A compression-diffusion-permutation strategy for securing image. *Signal Process.* **2018**, *150*, 183–190, doi:10.1016/j.sigpro.2018.04.014.
2. Wang, B.; Zhang, Q.; Wei, X. Tabu variable neighborhood search for designing DNA barcodes. *IEEE Trans. NanoBiosci.* **2020**, *19*, 127–131.
3. Li, X.; Wang, B.; Lv, H.; Yin, Q.; Zhang, Q.; Wei, X. Constraining DNA Sequences With a Triplet-Bases Unpaired. *IEEE Trans. NanoBiosci.* **2020**, *19*, 299–307, doi:10.1109/tnb.2020.2971644.
4. Hu, T.; Ouyang, C.-J.; Liu, Y.; Gong, L.-H. An image encryption scheme combining chaos with cycle operation for DNA sequences. *Nonlinear Dyn.* **2016**, *87*, 51–66, doi:10.1007/s11071-016-3024-6.
5. Chai, X.; Chen, Y.; Broyde, L. A novel chaos-based image encryption algorithm using DNA sequence operations. *Opt. Lasers Eng.* **2017**, *88*, 197–213, doi:10.1016/j.optlaseng.2016.08.009.
6. Liu, H.; Zhao, B.; Huang, L. Aremote-sensing image encryption scheme using dna bases probability and two-dimensional logistic map. *IEEE Access* **2019**, *7*, 65450–65459.
7. Enayatifar, R.; Guimarães, F.G.; Siarry, P. Index-based permutation-diffusion in multiple-image encryption using DNA sequence. *Opt. Lasers Eng.* **2019**, *115*, 131–140, doi:10.1016/j.optlaseng.2018.11.017.
8. Belazi, A.; Talha, M.; Kharbech, S.; Xiang, W. Novel Medical Image Encryption Scheme Based on Chaos and DNA Encoding. *IEEE Access* **2019**, *7*, 36667–36681, doi:10.1109/access.2019.2906292.
9. Huo, D.; Zhou, D.-F.; Yuan, S.; Yi, S.; Zhang, L.; Zhou, X. Image encryption using exclusive-OR with DNA complementary rules and double random phase encoding. *Phys. Lett. A* **2019**, *383*, 915–922, doi:10.1016/j.physleta.2018.12.011.
10. Revathy, K.; Thenmozhi, K.; Amirtharajan, R.; Praveenkumar, P. CR Assisted IE Guarded Authenticated Biomedical Image Transactions. *IEEE Photon. J.* **2018**, *10*, 1–13, doi:10.1109/jphot.2018.2872160.
11. Wang, X.; Hou, Y.; Wang, S.-B.; Li, R. A New Image Encryption Algorithm Based on CML and DNA Sequence. *IEEE Access* **2018**, *6*, 62272–62285, doi:10.1109/access.2018.2875676.
12. Chen, J.-X.; Zhu, Z.-L.; Zhang, L.-B.; Zhang, Y.; Yang, B.-Q. Exploiting self-adaptive permutation–diffusion and DNA random encoding for secure and efficient image encryption. *Signal Process.* **2018**, *142*, 340–353, doi:10.1016/j.sigpro.2017.07.034.
13. Liu, Z.; Wu, C.; Wang, J.; Hu, Y. A Color Image Encryption Using Dynamic DNA and 4-D Memristive Hyper-Chaos. *IEEE Access* **2019**, *7*, 78367–78378, doi:10.1109/access.2019.2922376.
14. Banu, S.A.; Amirtharajan, R. A robust medical image encryption in dual domain: Chaos-DNA-IWT combined approach. *Med Biol. Eng.* **2020**, *58*, 1445–1458, doi:10.1007/s11517-020-02178-w.
15. Ballesteros, D.M.; Peña, J.; Renza, D. A Novel Image Encryption Scheme Based on Collatz Conjecture. *Entropy* **2018**, *20*, 901, doi:10.3390/e20120901.
16. Ouyang, X.; Luo, Y.; Liu, J.; Cao, L.; Liu, Y. A color image encryption method based on memristive hyperchaotic system and DNA encryption. *Int. J. Mod. Phys. B* **2020**, *34*, 2050014, doi:10.1142/s0217979220500149.
17. Zhu, X.; Liu, H.; Liang, Y.; Wu, J. Image encryption based on Kronecker product over finite fields and DNA operation. *Optik* **2020**, 164725, doi:10.1016/j.ijleo.2020.164725.
18. Zhu, S.; Zhu, C. Secure Image Encryption Algorithm Based on Hyperchaos and Dynamic DNA Coding. *Entropy* **2020**, *22*, 772, doi:10.3390/e22070772.

19. Zhan, K.; Jiang, W. Novel four-wing hyper-chaos system and its application in image encryption. *Comput. Eng. Appl.* **2017**, *53*, 36–44.
20. Watson, J.D.; Crick, F.H.C. Molecular Structure of Nucleic Acids: A Structure for Deoxyribose Nucleic Acid. *Nature* **1953**, *171*, 737–738, doi:10.1038/171737a0.
21. Wang, X.; Wang, Y.; Zhu, X.; Luo, C. A novel chaotic algorithm for image encryption utilizing one-time pad based on pixel level and DNA level. *Opt. Lasers Eng.* **2020**, *125*, 105851, doi:10.1016/j.optlaseng.2019.105851.
22. Jithin, K.; Sankar, S. Colour image encryption algorithm combining Arnold map, DNA sequence operation, and a Mandelbrot set. *J. Inf. Secur. Appl.* **2020**, *50*, 102428, doi:10.1016/j.jisa.2019.102428.
23. Wu, X.; Zhu, B.; Hu, Y.; Ran, Y. A novel color image encryption scheme using rectangular transform-enhanced chaotic tent maps. *IEEE Access* **2017**, *5*, 6429–6436.
24. Álvarez, G.; Li, S. SOME BASIC CRYPTOGRAPHIC REQUIREMENTS FOR CHAOS-BASED CRYPTOSYSTEMS. *Int. J. Bifurc. Chaos* **2006**, *16*, 2129–2151, doi:10.1142/s0218127406015970.
25. Khan, S.; Han, L.; Lu, H.; Butt, K.K.; Bachira, G.; Khan, N. A new hybrid image encryption algorithm based on 2D-CA, FSM-DNA rule generator, and FSBI. *IEEE Access* **2019**, *7*, 81333–81350.
26. Khan, J.S.; Boulila, W.; Ahmad, J.; Rubaiee, S.; Rehman, A.U.; AlRoobaea, R.; Buchanan, W.J. DNA and Plaintext Dependent Chaotic Visual Selective Image Encryption. *IEEE Access* **2020**, *8*, 159732–159744, doi:10.1109/access.2020.3020917.
27. Zhang, Y.; Xiao, D. Self-adaptive permutation and combined global diffusion for chaotic color image encryption. *AEU-Int. J. Electron. Commun.* **2014**, *68*, 361–368, doi:10.1016/j.aeue.2013.10.002.
28. Rehman, A.U.; Liao, X.; Ashraf, R.; Ullah, S.; Wang, H. A color image encryption technique using exclusive-OR with DNA complementary rules based on chaos theory and SHA-2. *Optik* **2018**, *159*, 348–367, doi:10.1016/j.ijleo.2018.01.064.
29. Chai, X.; Fu, X.; Gan, Z.; Lu, Y.; Chen, Y. A color image cryptosystem based on dynamic DNA encryption and chaos. *Signal Process.* **2019**, *155*, 44–62, doi:10.1016/j.sigpro.2018.09.029.
30. Wang, X.-Y.; Zhang, H.-L.; Bao, X.-M. Color image encryption scheme using CML and DNA sequence operations. *Biosystems* **2016**, *144*, 18–26, doi:10.1016/j.biosystems.2016.03.011.
31. Qayyum, A.; Ahmad, J.; Boulila, W.; Rubaiee, S.; Arshad; Masood, F.; Khan, F.; Buchanan, W.J. Chaos-based Confusion and Diffusion of Image Pixels using Dynamic Substitution. *IEEE Access* **2020**, *8*, 1, doi:10.1109/access.2020.3012912.
32. Masood, F.; Boulila, W.; Ahmad, J.; Arshad, A.; Sankar, S.; Rubaiee, S.; Buchanan, W.J. A Novel Privacy Approach of Digital Aerial Images Based on Mersenne Twister Method with DNA Genetic Encoding and Chaos. *Remote Sens.* **2020**, *12*, 1893, doi:10.3390/rs12111893.